UNITED STATES PATENT APPLICATION

FOR

SUPPORT INTERFACE MODULE BUG SUBMITTER

INVENTORS:

Maarten W. 't Hooft, a citizen of the Netherlands
Ethan J. Rider, a citizen of the United States

ASSIGNED TO:

Sun Microsystems, Inc., a California Corporation

PREPARED BY:

THELEN, REID, & PRIEST
P.O. BOX 640640
SAN JOSE, CA 95164-0640
TELEPHONE: (408) 292-5800
FAX: (408) 287-8040

Attorney Docket Number: SUN-P7270

Client Docket Number: P7270

SPECIFICATION

TITLE OF THE INVENTION

SUPPORT INTERFACE MODULE BUG SUBMITTER

CROSS-REFERENCES

[0001] This is a continuation-in-part of U.S. Patent Application Serial No. 09/___,___, filed November 9, 2001, in the name of inventor Maarten W. 't Hooft, entitled "Support Interface Module", commonly assigned herewith.

FIELD OF THE INVENTION

[0002] The present invention relates generally to the field of computer science. More specifically, the present invention relates to a support interface module bug submitter over which channels of information can flow between an application interface and an external support service.

BACKGROUND OF THE INVENTION

[0003] Despite the advancement of technology, users continue to experience problems with computer systems. Support may then be needed. Support may include a number of services including an explanation of features or the trouble shooting of problems. Traditional support takes many forms. For instance, support may take the form of a user manual. Many users find these little help at all if they can even locate the manual when they need one. To combat this, support may take the form of an embedded help function. This may be query or menu driven.

Again this is a form of self-help and it is limited to the time when the function is fixed in the application. To reduce time dependency and memory demands, support may take the form of a dedicated help server external to the application. Increasingly, applications and users are connected together by networks such as the Internet. In this case, the user leaves the application and, via the network, contacts the help server independently and seeks the support that they need there. If this does not prove adequate, the user may phone a dedicated help provider and speak with one of their representatives or listen to their automated service. Individually or in combination, traditional forms of support may often prove unsatisfactory to the user.

[0004] A bug is a persistent error in a software or hardware. One way to get rid of the bug it to modify the program, usually with a software patch for software. A bug report consisting of data describing the error that occurred, may be submitted to a support provider or the software developer. The support provider once notified of the bug, may then prepare a software patch or a new version of the software removing the bugs previously found.

[0005] A definite need exists for a bug submission module allowing the user to select how and where a bug report is submitted. A primary purpose of the present invention is to solve these needs and provide further related advantages.

BRIEF DESCRIPTION OF THE INVENTION

[0006] A method for submitting a bug report utilizes a bug submission module to request a bug submission service from a first support host using a Support Interface Module for communicating with the first support host. The bug submission service includes a list of data to be collected and the return address of a second support host. The first support host also includes a support services resource. The bug submission module receives the requested bug submission service from the first support host using the Support Interface Module and collects data based on the list of data to be collected. The bug submission module then sends the collected data to the return address of second support host using the Support Interface Module.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] The accompanying drawings, which are incorporated into and constitute a part of this specification, illustrate one or more exemplary embodiments of the present invention and, together with the detailed description, serve to explain the principles and exemplary implementations of the invention.

[0008] In the drawings:

FIG. 1 is a block diagram that illustrates a Support Interface Module bug submission system according to a specific embodiment of the present invention;

FIG. 2 is a block diagram that illustrates a bug submission module and the Support Interface Module of FIG. 1 according to a specific embodiment of the present invention; and

FIG. 3 is a flow diagram that illustrates a method for a bug submission according to a specific embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0009] Embodiments of the present invention are described herein in the context of a support interface module bug submitter. Those of ordinary skill in the art will realize that the following detailed description of the present invention is illustrative only and is not intended to be in any way limiting. Other embodiments of the present invention will readily suggest themselves to such skilled persons having the benefit of this disclosure. Reference will now be made in detail to exemplary implementations of the present invention as illustrated in the accompanying drawings. The same reference indicators will be used throughout the drawings and the following detailed descriptions to refer to the same or like parts.

[0010] In the interest of clarity, not all of the routine features of the exemplary implementations described herein are shown and described. It will of course, be appreciated that in the development of any such actual implementation, numerous implementation-specific decisions must be made in order to achieve the developer's specific goals, such as compliance with application- and business-related constraints, and that these specific goals will vary from one implementation to another and from one developer to another. Moreover, it will be appreciated that such a development effort might be complex and time-consuming, but would nevertheless be a routine undertaking of engineering for those of ordinary skill in the art having the benefit of this disclosure.

[0011] In accordance with one embodiment of the present invention, the components, process steps, and/or data structures may be implemented using various types of operating systems, computing platforms, computer programs, and/or general purpose machines. In

addition, those of ordinary skill in the art will recognize that devices of a less general purpose nature, such as hardwired devices, field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), or the like, may also be used without departing from the scope and spirit of the inventive concepts disclosed herein.

[0012] Turning first to FIG. 1, a block diagram of a bug submission system 100 is shown. The system 100 includes a host system 102, a first support host 104, and a network 106 connecting the host system 102 to the first support host 104. One of ordinary skill in the art will recognize that the network 106 may take one of many forms. For the sake of clarity, these will not be presented here. The host system 102 includes at least one application having an interface 108 for the user. The application may also take many forms including word processor or network browser. Likewise, the outward manifestation of the interface 108 may take many forms but will typically be a graphical user interface. The interface 108 includes a bug submission module 110 that enables the user of the application to submit a bug report for a particular module 111 having a bug. The bug submission module 110 is described in more detail below. In addition, the bug submission module 110 communicates with and through a support interface module (SIM) 112. The SIM 112 may be integral to the application, to the host system 102, or to some combination of both. In some contexts, the SIM 112 may not be referred to as a module but will still perform the same functions. The SIM 112 provides the infrastructure over which channels of information can flow between the host system 102 and the first support host 104. The first support host 104 includes a support services resource 114 that is called upon to help submit the bug. In one specific embodiment, the support services resource 114 includes a bug submission service having a list of data to be collected on the host system 102 and the return

address of the support host that the list of collected data will be sent to when that service is selected. The collected data can either be sent to the first support host 104 or to a third party support host, such as second support host 116. The destination of the collected data depends on the user's choice of service. The user should only view the result of the collaboration between the bug submission module 110 and the SIM 112. The result is a bug submission presence within the application itself.

[0013] In order for the SIM 112 to configure and manage the communication of data from the bug submission module 110 to the first support host 104, the SIM 112 may need information such as the address of the support services resource 114, the type of communication protocol to be used, the port to contact and the eligibility of use of the support services resource 114. The SIM 112 can obtain the necessary information in at least one of three different ways. Two of these methods rely on the use of description facilities while the third requires no description searching.

[0014] According to one embodiment, the SIM 112 can contact a support services registry for information on a service. The location of the support services registry can be predetermined or an address for the registry can be provided by an outside source. Among other information, the registry will contain information on whether a service exists, where it is located, and who may use it.

[0015] According to another embodiment, the SIM 112 may contact a service description servlet at the instruction of the bug submission module 110. The SIM 112 looks in a location

specified by the bug submission module 110 for information regarding a particular service. The service description servlet will then pass on the information in a format similar to the one found on the support services registry.

[0016] According to a yet another embodiment, the SIM 112 may, on occasion, find that the bug submission module 110 itself contains all of the information needed. If so, the SIM 112 utilizes the information provided by the bug submission module 110 without reference to a registry or description service.

[0017] There are a number of general attributes that one might want the SIM 112 to exhibit. If one or more of these attributes are contrary to one another or to the specific environment, then they can be deleted or modified. One would prefer that the architecture of the SIM 112 be able to provide seamless communication between the user and the support services resource 114. Further, one would prefer that the SIM 112 be aware of the networking environment in which it resides. This awareness might include knowing which ports are accessible, what the various latencies are, what bandwidth is available, and what the current traffic load is. Further still, one would prefer that the SIM 112 be able to ensure the security and integrity of all of its communications with the first support host 104. Either continuously or on demand, one would prefer that the user be able to monitor the activity of the SIM 112. One would also prefer that the SIM 112 be able to correct for failed communication links or at least be able to present the user with options in such an event. The SIM 112 may communicate with one or more of any number of communications protocols in either or both a synchronous and an asynchronous mode.

[0018] Turning now to FIG. 2, a block diagram of the bug submission module 202 and the Support Interface Module 204 of FIG. 1 is shown. The SIM 204 is shown to include a session handler 206, at least one session 208, a transport handler 210, and at least one transport 212 for each session 208. Each component plays a role in the process of communication between the bug submission module 202 and the support services resource 114 of FIG. 1. Each component may be independent of one another or the SIM 202. Conversely, the function of each component may be combined in whole or in part without departing from the inventive concept disclosed herein. The session handler 206 provides coordination of the one or more sessions 208 that are ongoing. The transport handler 210 provides coordination of the one or more transports 212 per ongoing session 208.

[0019] One example of a communication process by the bug submission module 202 assumes that a user makes a request, then the bug submission module 202 initially processes this request. It may be the case that the bug submission module 202 can handle the request alone. When the bug submission module 202 can not handle all or some portion of the request alone, then it contacts the SIM 204. The SIM 204 passes the request to the session handler 206 which approves the request and generates at least one session 208 for the request. The session 208 in turn initializes the transport handler 210 for the communication required to deal with the request. The transport handler 210 generates at least one suitable transport 212 for the session 208. The transport 212 sends data to and/or receive data from the support services resource 114 of FIG. 1. It may be the case that the transport handler 210 is instructed to redirect the communication. In such a case, the transport handler 210 may generate a new transport 212 and terminate the

original transport 212. Through its API, the session 208 cooperates with the bug submission module 202 to satisfy the user's request. Consequently, after the initial request, all subsequent communication by the bug submission module 202 for the request is with the session 208 and not with the session handler 206. When the request of the user is either satisfied or withdrawn, some or all of the sessions 208 and the transports 212 that were generated for that request may be terminated. Multiple requests may be processed in sequence or in parallel.

[0020] As above with the SIM 204 in general, there are a number of general attributes that one might want the components of the SIM 204 to exhibit. If one or more of these attributes are contrary to one another or to the specific environment, then they can be deleted or modified. For example, one would expect that the session handler 206 would be aware of all of the sessions 208 that are generated. Further, one would prefer that the session handler 206 hold data about the host system 102 and the network 106 of FIG. 1. Also, one would prefer that the session handler 206 be aware of the network activity originating from the host system 202. In addition, one would prefer that the session handler 206 use this network activity awareness to scale its own network activity to avoid significant performance loss to the host system 102. It would also be preferred that the session handler 206 to be able to provide information regarding or correction of communication failures. Further still, one would prefer that the session handler 206 be able to provide status info for each session 208 either continuously or on demand.

[0021] It may be the case that a host system 102 includes multiple SIMs 204. In such a case, it may be necessary to coordinate the individual SIMs 204. This may be accomplished in a number of ways known to one of ordinary skill including disabling all but one of the session

handlers 206 and placing the non-disabled session handler 206 in control of all of the SIMs 204. Another technique would be for the multiple session handlers 206 to hold an election to determine which session handler 206 will control the multiple SIMs 204.

[0022] With respect to the sessions 208, although there may be multiple sessions 208 running in series or in parallel, the individual sessions 208 may not be aware of one another. Further, one would prefer that the sessions 208 have a finite length and that either or both the session handler 206 and the bug submission module 202 be able to terminate the session 208. One would prefer for the session 208 to control communications between the SIM 204 and the bug submission module 202.

[0023] With respect to the transport handler 210, one would expect that the transport handler 210 would be aware of all of the transports 212 that it has generated. If there are multiple SIMs 204, a first transport handler 210 may or may not be aware of a second transport handler 210 or the transports 212 that the second transport handler 210 has generated. Further, one would expect the transport handler 210 to be able to communicate using one or more of any number of protocols. One would prefer that the transport handler 210 be able to generate diagnostic information on the network environment and characteristics and be able to pass this information on to the session handler 206. One would also prefer that the transport handler 210 be able to report errors and exceptions to the session handler 206.

[0024] With respect to the transports 212, although there may be multiple transports 212 in series or in parallel, the individual transports 212 may not be aware of one another. It may be

the case that the transport 212 is channel dependent such that when the channel fails the transport 212 ends and a new transport 212 may have to be generated by the transport handler 210 to complete the failed communication.

[0025] According to one embodiment of the present invention, the Bug Submission Module 202 provides the user with several options. For instance, prior to submitting the bug, the user is presented with an opportunity to troubleshoot the problem (which will also automate the problem description), and review and approve any data collected. If the user chooses to submit the problem, an acknowledgement of the submission is provided. Once the context of the bug (the relevant module) is known, the Bug Submission module 202 presents the SIM services available for that module. Thus the SIM 204 may be used to invoke other arbitrary SIM services in the process of submitting a bug. This allows for more features to be embedded dynamically into the Bug Submitter module 202.

[0026] The Bug Submitter Module 202 submits its bug report through the SIM 112 and network 106. One of ordinarily skill in the art will realize that there are many ways in which the SIM 112 may communicate with network 106. For example, SIM 112 may submit the bug reports via SOAP (over HTTP and SMTP). Additionally, the bug submission module 202 may provide an API for module developers to use their own transport mechanisms. According to one embodiment, the bug submission module 202 may not use SOAP but may simply open a Socket connection to transport the data to a support host. This would allow the use of third party toolkit vendors for an easy and extensible mechanism for transporting data. Since SOAP messaging is

not bi-directional, the Bug Submission module 202 may rely on the SIM architecture to provide message acknowledgements where possible.

[0027] According to one embodiment, the bug submission module 202 comprises a receiver (not shown) for receiving a submission service using the SIM 112 to communicate with the first support host 104. The submission service comprises a list of data to be collected and the return address of a second support host. A collector (not shown) collects the data based on the list of data to be collected. A sender (not shown) sends the collected data to the second support host 116 using the SIM 112.

[0028] The bug submission module 202 may also incorporate a data collection service and a data delivery mechanism. The data collection service provides mechanisms for gathering information commonly requested by support engineers, as well as a mechanism for module developer to add data requirements for their reports. The data delivery mechanism may utilize a portion of a Support Services Interface Module API to transfer the collected data to a destination in the appropriate format.

[0029] According to one embodiment, with respect to the data collection service of the bug submission module 202, each report template should have specific information to collect. All automated data collectors may include a collection timeout. A call to retrieve collected data should either return the data when the collection is completed or set the data state to a failed status. The API for data collection should allow for module developers to provide custom report templates and data collection that follows the previously listed preferences. The collected data

may include, by way of example, requests for user supplied data such as steps to reproduce, or support contract information. Such user supplied data should distinguish empty input from a failure to acquire input. All data collected should be associated with a data presenter appropriate for the user environment.

[0030] According to one embodiment, with respect to the data submission mechanism of the bug submission module 202, the data may be submitted only to the destination accepted by the user. Such destination should have at least one transport. However in case of failure, more than one transport may be used. The transport may specify a protocol and address for the transmission of the data. The destination may provide information to prove its identity to the user. The destination may attempt to use alternate transports prior to failing completely. The destination may also attempt to confirm transmission. In the event of a complete transmission failure, the bug submission module 202 should allow for retransmission or the specification of a new destination.

[0031] Turning now to FIG. 3, a flow diagram of a method for submitting a bug report according to the bug submission module 202 is shown. In a first decision block 302, a bug is detected on the host system 102 of FIG. 1. When a bug is detected, the bug submission module 110 of FIG. 1 receives a notification about the presence of the bug. The Bug Submission module 110 then requests a Bug Submission service from a first support host 104 in block 304. In response to the request, the requested Bug Submission service 110 is received in block 306. The Submission Service comprises a list of data to be collected and the return address of a support host. In block 308, the bug submission module 110 collects data based on the list of data to be

collected as specified in the requested submission service. The bug submission module 110 submits the collected data via SIM 112 to the support host as specified in the submission service in block 310. The support host may either be the first support host 104 or another support host such as second support host 116.

[0032] According to another embodiment, the SIM may provide for a Collector class (not shown) to developers. Collectors are intended to obtain data from the user or the environment. Those collectors that retrieve information from the environment (the user's system) will be required to be able to run more than once, such that if a user accidentally overwrites the collected data, it can be retrieved once again. If it is not possible for the value to change between runs, it is acceptable for the Collector to simply cache and return the last value obtained.

[0033] According to another embodiment of the present invention, after all data collection has been done, each piece of data will be presented to the user. Each piece of data will be associated with a Presenter, either because the data has a known MIME type associated with a presenter, or because the collector has associated this piece of data with a specific Presenter instance. It is a design goal to keep this API generic enough to be used from the command line.

[0034] One feature of the SIM bug submission module is the ability for third parties (Support Providers) to leverage the features for their customer base and support staff. Developer support may also include example codes and extensive base classes where applicable.

[0035] According to another embodiment, all bug reports (regardless of the report structure) must be compatible with a to-disk Transport provided by the SIM bug submission module. Basically this means that the bug report must be able to be saved to a file. The files should be designed so that they can be sent using a different user and machine than that which generated the report.

[0036] While embodiments and applications of this invention have been shown and described, it would be apparent to those skilled in the art having the benefit of this disclosure that many more modifications than mentioned above are possible without departing from the inventive concepts herein. The invention, therefore, is not to be restricted except in the spirit of the appended claims.